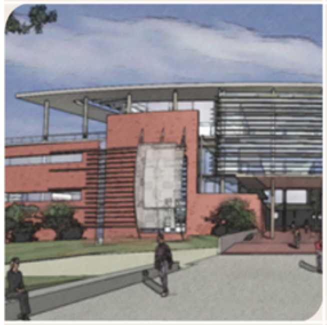
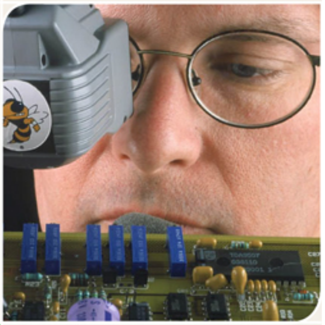


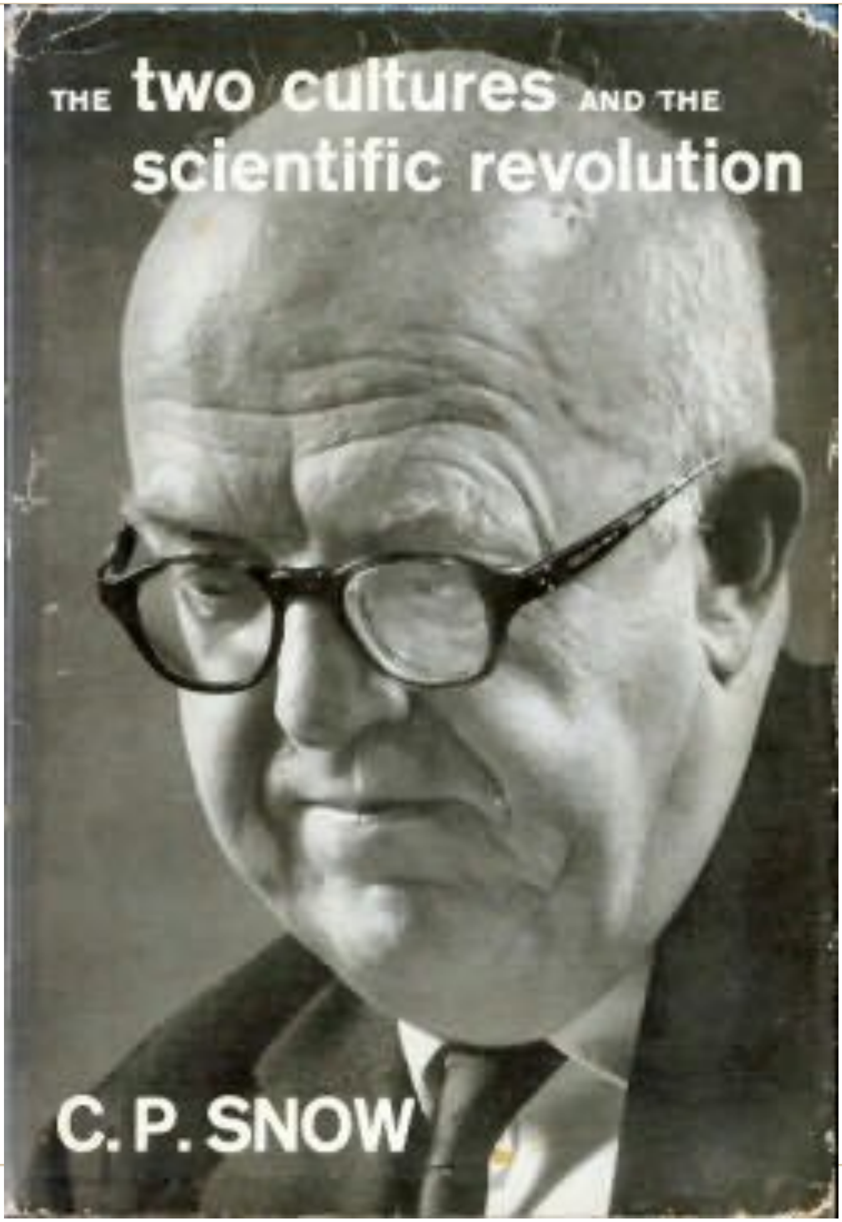


Helping Everyone Create with Computing

Mark Guzdial
School of Interactive Computing

Georgia Tech College of Computing

The Two Cultures



Story

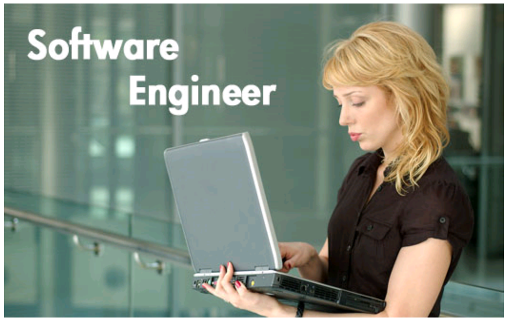
- Computing is important for more than just those who choose to major in computing.
  - Who are those (the majority) who need computing but won’t major in computing? What do they need from CS, and why aren’t in CS classes? How do they now learn CS?

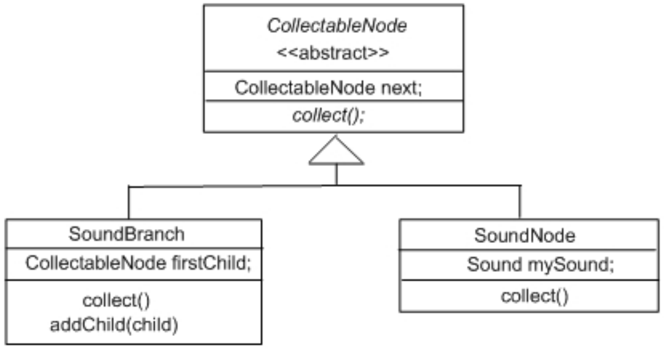
- **It is the job of computer scientists to teach everyone to create with computer science, explicitly, *programming*.**

- How do we teach CS to those who do *not* want to become software engineers or computer scientists?
  - Story #1: The Story of Computing for All at Georgia Tech.
    - How we’ve used and failed at on-line learning
  - Story #2: “Teaching” Graphics Designers about Computer Science.
  - Story #3: Understanding the Needs of High School Teachers Learning Computer Science

The typical CS student: Future Software Engineer

- To produce reliable, robust, secure software.
- To work in interdisciplinary teams.
- To use appropriate design notations, such as UML.
- To work in multiple programming languages.

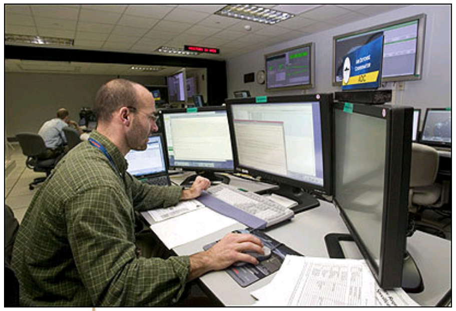




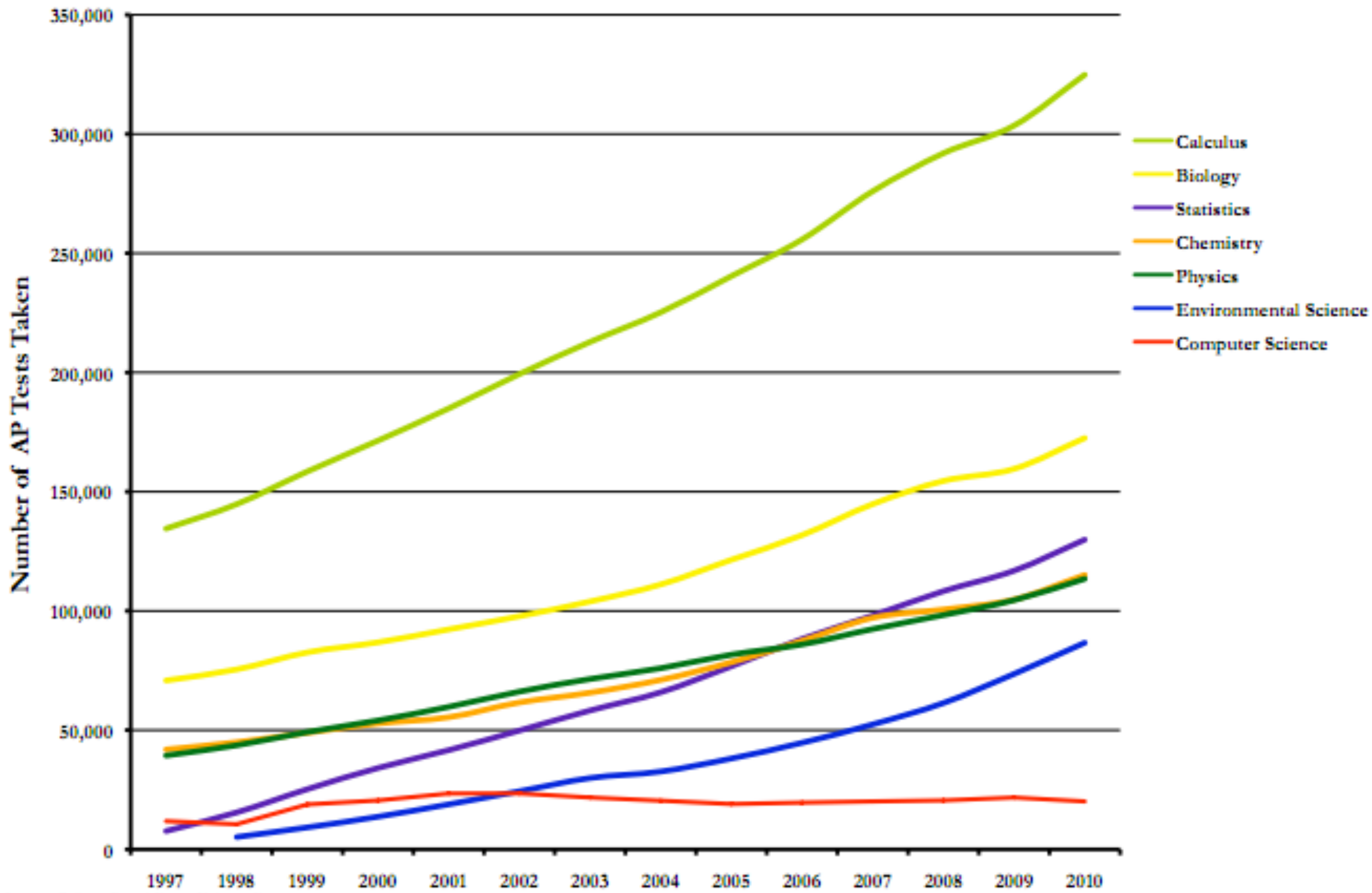
CollectableNode
<<abstract>>
CollectableNode next;
collect();

SoundBranch
CollectableNode firstChild;
collect()
addChild(child)

SoundNode
Sound mySound;
collect()



High School Participation in AP STEM Disciplines



Source: College Board Exam Volume Data

*Computer Science up until 2009 had an A and AB test as of 2010 they are only offering the A test

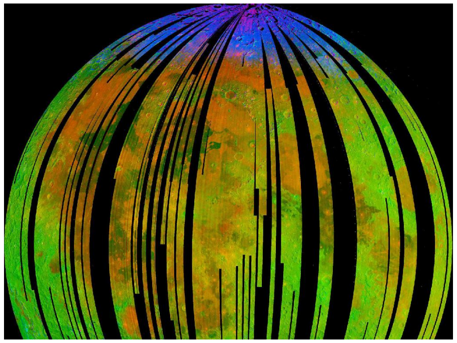
— Chris Stephenson, CSTA, 2010

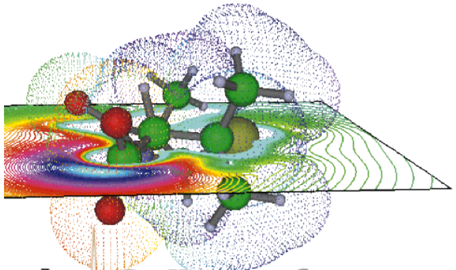
Who else needs or wants what CS has to offer?

- Computing is at the core of the modern society and modern economy.
- Computing is key to innovation in many disciplines.
- ***Computer Science has a much larger potential audience beyond software developers.***
  - Estimates:
    ~13 million non-professional programmer/end-user programmers in US by 2012,
    vs.
    ~3 million professional software developers (Scaffidi, Shaw, & Myers, 2005)

An atypical CS student:
Future computational scientist or engineer

- To use computation as a tool to enhance understanding.
- To write programs of (at most) 100 lines (most often, 10 lines) for themselves.
  - They care about the products of the programs, not the programs.
- To learn as few languages as are needed for their tasks.
- To work in interdisciplinary teams, including software engineers.





An atypical CS student: Future high school CS teacher

- To use code to explore and understand ideas of computation.
- To learn what languages are necessary to meet standards and engage students.
- To work with students with a wide range of interests.
    - Probably won’t work with professional software engineers

An atypical CS student: Future graphics designer

- To write programs to improve their efficiency, and to implement their dynamic (e.g., Web) designs.
- To do as little coding as possible.
- To learn about computing ideas in order to improve their process, but with a focus on people and creativity.
  - Probably won't work with professional software engineers

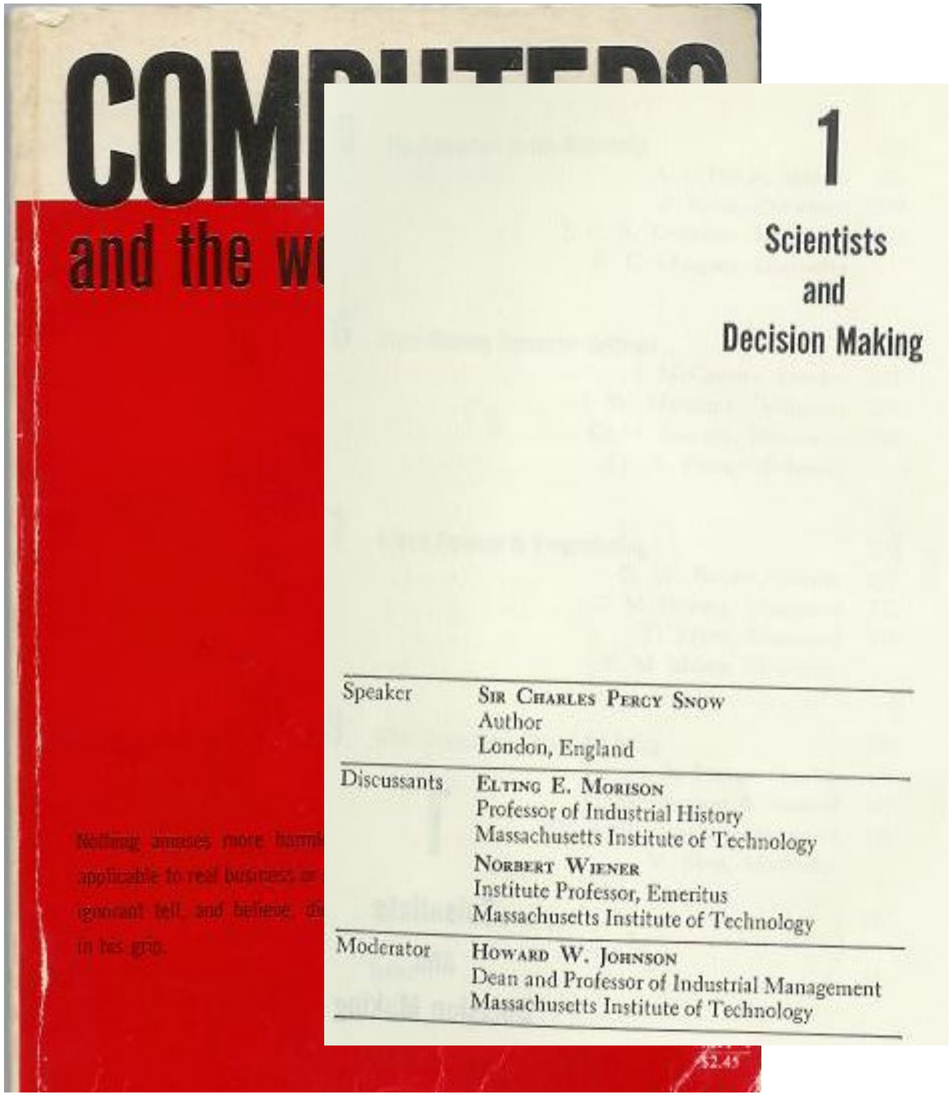
How do meet this need?

- Our track record for the first CS course is poor.
  - 30-50% failure or withdrawal rates (Bennedsen & Caspersen, 2007)
- Other majors tend to be more female and more ethnically diverse than the typical computing student.
  - Our track record with these audiences is particularly poor (Margolis & Fisher, 2003)

Us?

- Is it the job of computer science to teach computer science to everyone?
- Do *they* want what we have to offer?

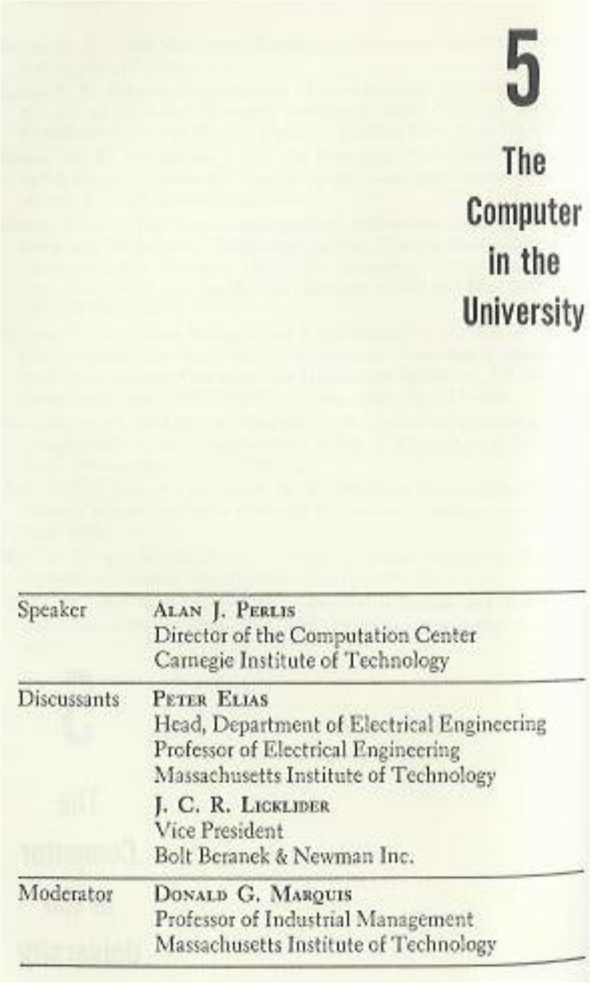
1961 MIT Sloan School Symposium



1

Scientists and Decision Making

Speaker	SIR CHARLES PERCY SNOW Author London, England
Discussants	ELTING E. MORISON Professor of Industrial History Massachusetts Institute of Technology NORBERT WIENER Institute Professor, Emeritus Massachusetts Institute of Technology
Moderator	HOWARD W. JOHNSON Dean and Professor of Industrial Management Massachusetts Institute of Technology



5

The Computer in the University

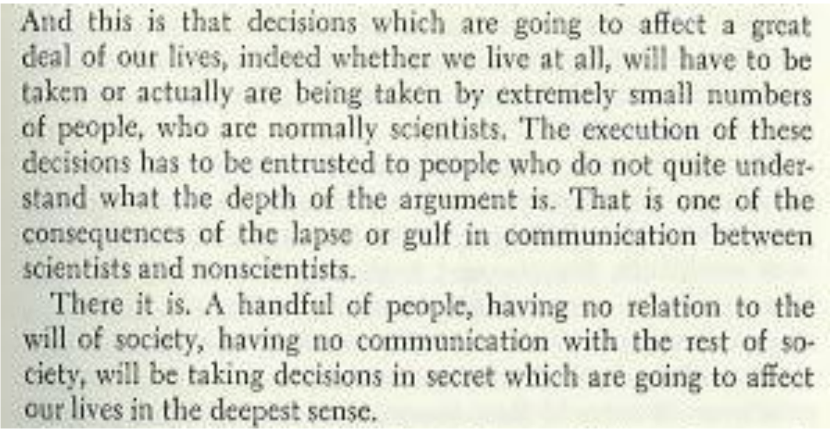
Speaker	ALAN J. PERLIS Director of the Computation Center Carnegie Institute of Technology
Discussants	PETER ELIAS Head, Department of Electrical Engineering Professor of Electrical Engineering Massachusetts Institute of Technology J. C. R. LICKLIDER Vice President Bolt Beranek & Newman Inc.
Moderator	DONALD G. MARQUIS Professor of Industrial Management Massachusetts Institute of Technology

Learn Programming to Re-Think Process Everywhere

- Alan Perlis argued that computer science should be part of a liberal education.
  - Explicitly, he argued that all students should learn to program.
- Why?
  - Because Computer Science is the study of process.
  - Automated execution of process changes everything
    - Including how we think about things we already know

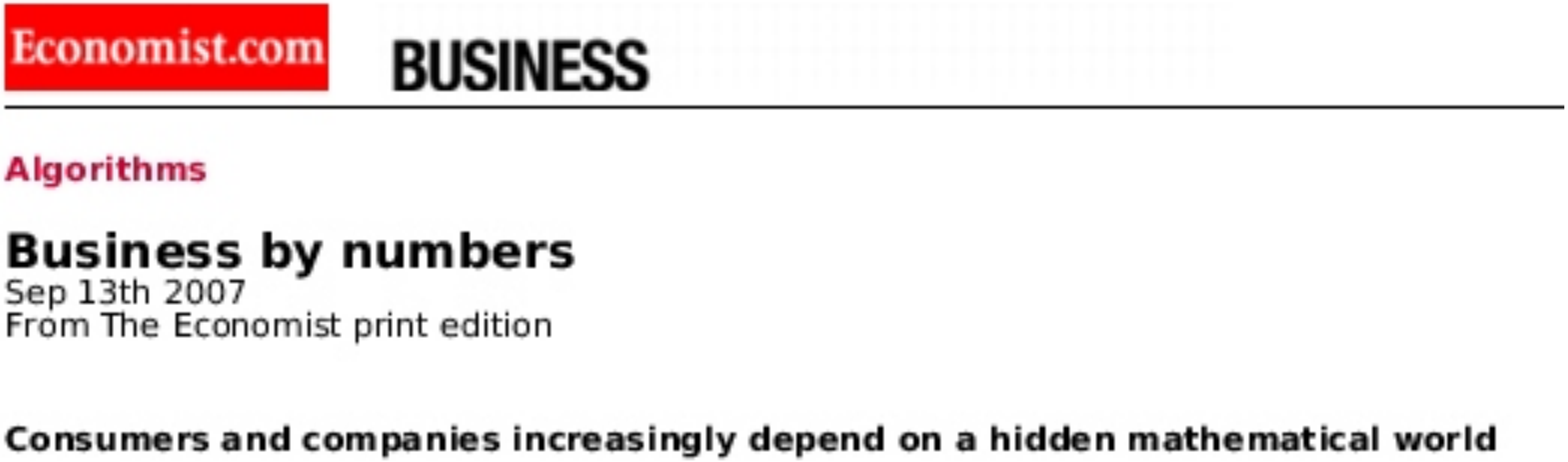
The Power and Fear of Algorithms

- The Economist (Sept., 2007) spoke to the algorithms that control us, yet we don’t understand.
  - Credit Ratings, Adjustable Rate Mortgages, Search Rankings



And this is that decisions which are going to affect a great deal of our lives, indeed whether we live at all, will have to be taken or actually are being taken by extremely small numbers of people, who are normally scientists. The execution of these decisions has to be entrusted to people who do not quite understand what the depth of the argument is. That is one of the consequences of the lapse or gulf in communication between scientists and nonscientists.

There it is. A handful of people, having no relation to the will of society, having no communication with the rest of society, will be taking decisions in secret which are going to affect our lives in the deepest sense.



Economist.com BUSINESS

Algorithms

Business by numbers

Sep 13th 2007
From The Economist print edition

Consumers and companies increasingly depend on a hidden mathematical world

Richard Dawkins on Biology as Computer Science

- On US *National Public Radio* in April 2007:

- GROSS: You close your book saying, "I am thrilled to be alive at a time when humanity is pushing against the limits of understanding." How do you think that's happening in your field of evolutionary biology?
- Mr. DAWKINS: Well, it's the most exciting time to be a biologist… Since Watson and Crick in 1953, biology has become a sort of branch of computer science. I mean, genes are just long computer tapes, and they use a code which is just another kind of computer code. It's quaternary rather than binary, but it's read in a sequential way just like a computer tape. It's transcribed. It's copied and pasted. All the familiar metaphors from computer science fit.

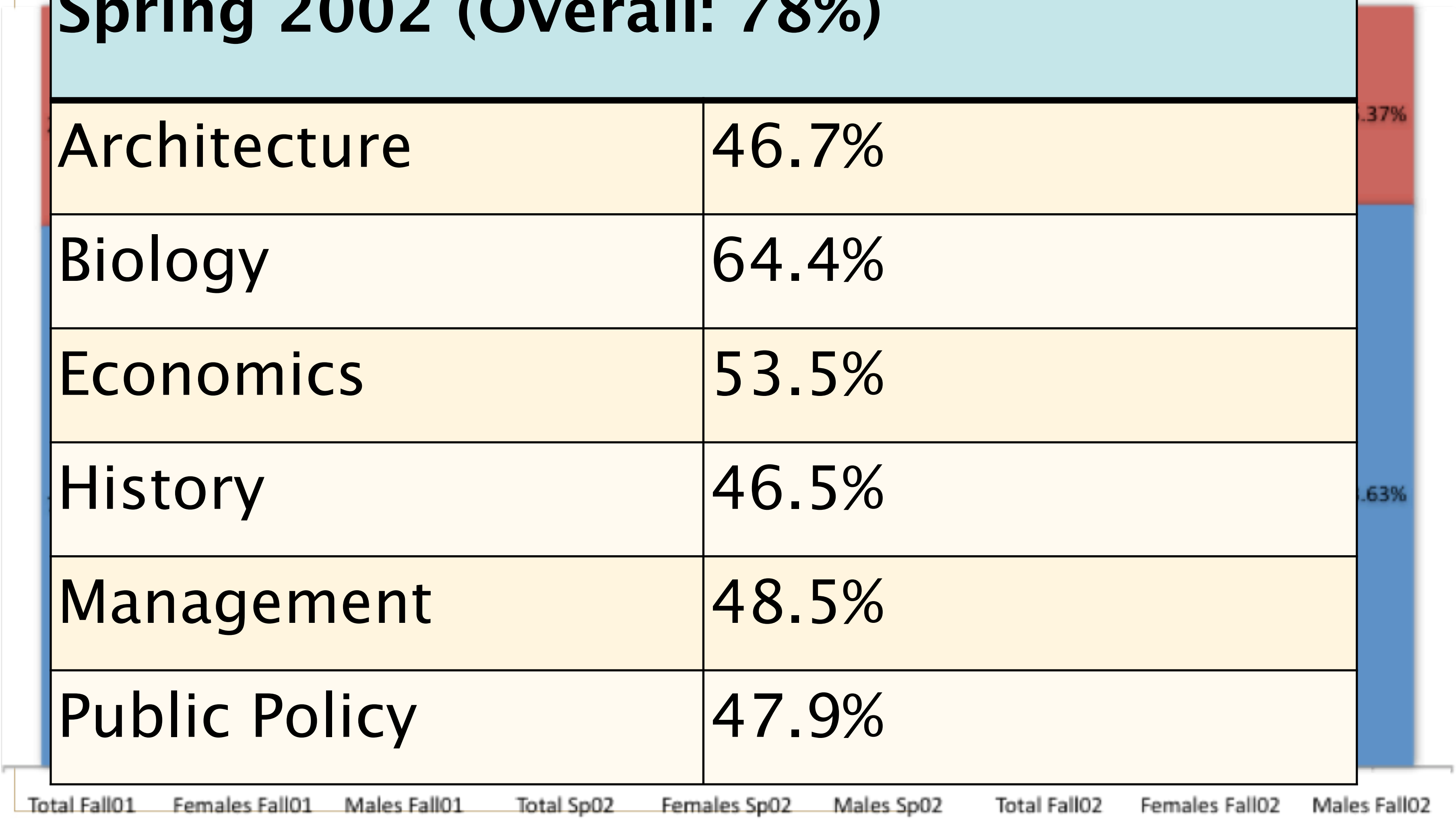
Three Non-Traditional Audiences

- Story #1: The Story of Computing for All at Georgia Tech.
- Story #2: “Teaching” Graphics Designers about Computer Science.
- Story #3: Understanding the Needs of High School Teachers Learning Computer Science

Story #1: Teaching Computing to Everyone

- *Fall 1999*:
  All students at Georgia Tech must take a course in computer science.
  - Considered part of General Education, like mathematics, social science, humanities…
- 1999-2003: Only one course met the requirement.
  - Shackelford's pseudocode approach in 1999
  - Later Scheme: How to Design Programs (MIT Press)

One-class CS1: Pass (A, B, or C) vs. WDF (Withdrawal, D or F)



Success Rates in CS1 from Fall 1999 to Spring 2002 (Overall: 78%)	
Architecture	46.7%
Biology	64.4%
Economics	53.5%
History	46.5%
Management	48.5%
Public Policy	47.9%

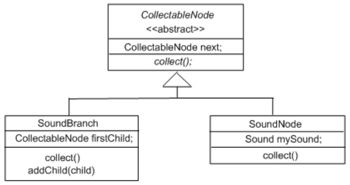
Contextualized Computing Education

- What's going on?
  - Research results: Computing is "tedious, boring, irrelevant"
- Since Spring 2003, Georgia Tech teaches three introductory CS courses.
  - Based on Margolis and Fisher's "alternative paths"
- Each course introduces computing using a context (examples, homework assignments, lecture discussion) relevant to majors.
  - Make computing relevant by teaching it in terms of what computers are good for (from the students' perspective)

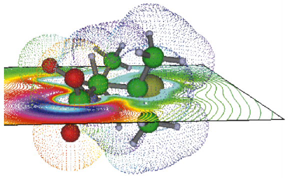
Unlocking the Clubhouse
Women in Computing
Jane Margolis and Allan Fisher
Tech-Savvy
Educating Girls in the New Computer Age
Preparing Women and Minorities for the IT Workforce
The Role of Nontraditional Educational Pathways
AAAS

Our Three CS1's Today

- CS1301/1321 *Introduction to Computing* Traditional CS1 for our CS majors and Science majors (math, physics, psychology, etc.). Now, uses robots.



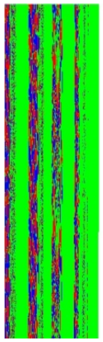
- CS1371 *Computing for Engineers* CS1 for Engineers. Same topics as CS1301, but using MATLAB with Engineering problems.



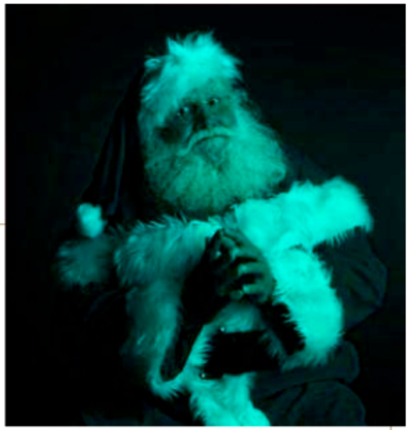
- CS1315 *Introduction to Media Computation* for Architecture, Management, and Liberal Arts students.

Media Computation: Teaching in a Relevant Context

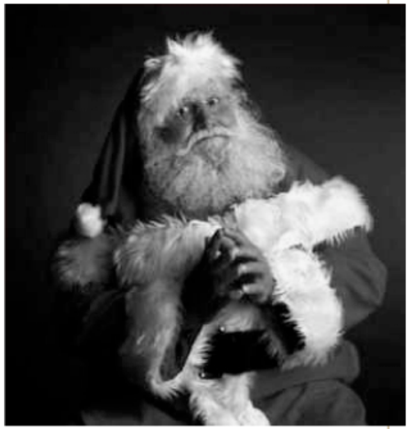
- Presenting CS topics with media projects and examples
  - Iteration as creating negative and grayscale images
  - Indexing in a range as removing redeye
  - Algorithms for blending both images and sounds
  - Linked lists as song fragments woven to make music
  - Information encodings as sound visualizations



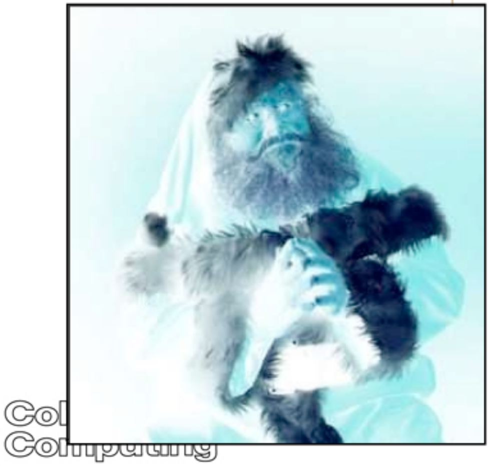
```
def clearRed(picture):
  for pixel in getPixels(picture):
    setRed(pixel,0)
```



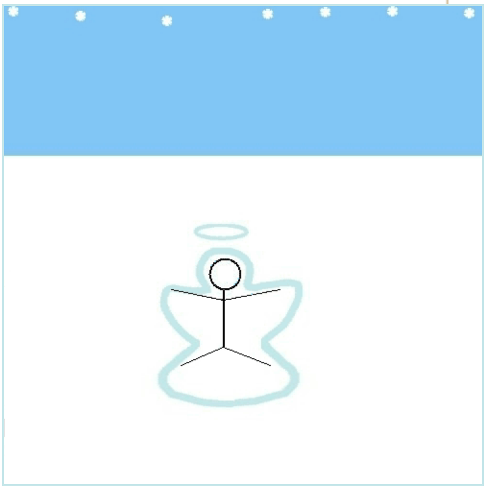
```
def greyscale(picture):
  for p in getPixels(picture):
    redness=getRed(p)
    greenness=getGreen(p)
    blueness=getBlue(p)
    luminance=(redness+blueness+greenness)/3
    setColor(p, makeColor(luminance,luminance,luminance))
```

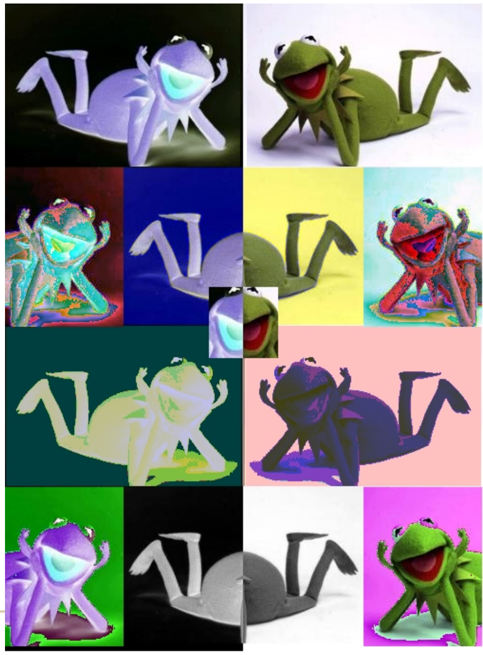


```
def negative(picture):
  for px in getPixels(picture):
    red=getRed(px)
    green=getGreen(px)
    blue=getBlue(px)
    negColor=makeColor(255-red,255-green,255-blue)
    setColor(px,negColor)
```



Open-ended, contextualized homework in Media Computation CS1 & CS2





Sound collage

Linked list as canon

Results:CS1“Media Computation”

Change in Success rates in CS1 “Media Computation” from Spring 2003 to Fall 2005 (Overall 85%)		
Architecture	*46.7%*	85.7%
Biology	*64.4%*	90.4%
Economics	*54.5%*	92.0%
History	*46.5%*	67.6%
Management	*48.5%*	87.8%
Public Policy	*47.9%*	85.4%

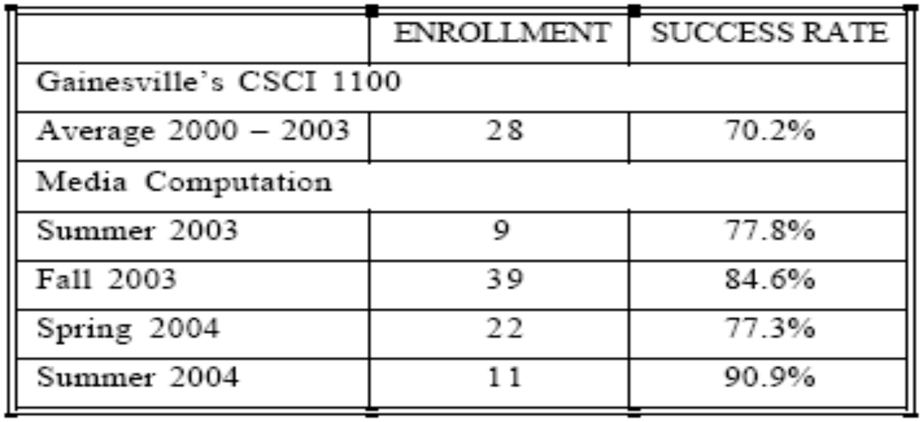
🟥 WDF

🟦 Pass

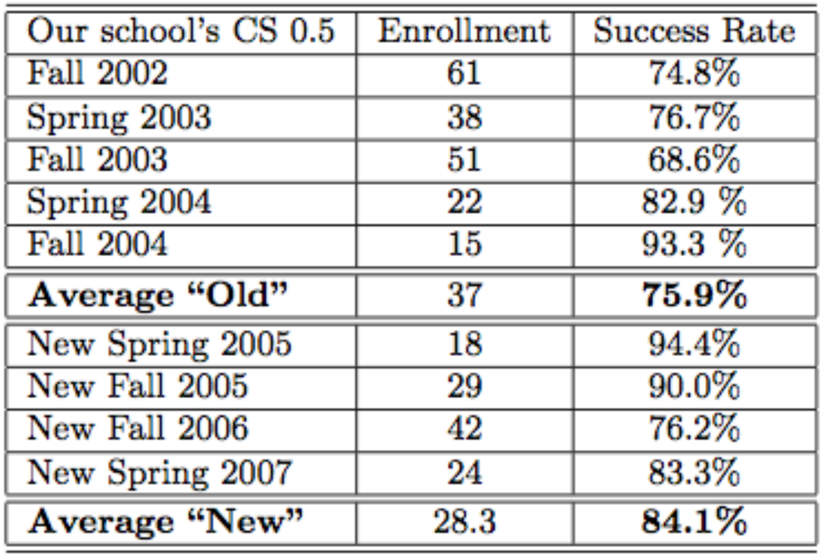
Results at Other Schools

- Similar retention results at 2 year public Gainesville College (Tew, Fowler, Guzdial, SIGCSE 2005) and at (much more diverse) U. Illinois-Chicago's CS0.5 (Sloan & Troy, SIGCSE 2008)
- Would you like more CS?
  - GT 15.2% "Strongly Disagree." <25% agree.
  - More MediaComp? GT and Gainesville over 40% agree.

Table 2. Success rates at Gainesville College before and with Media Computation class.



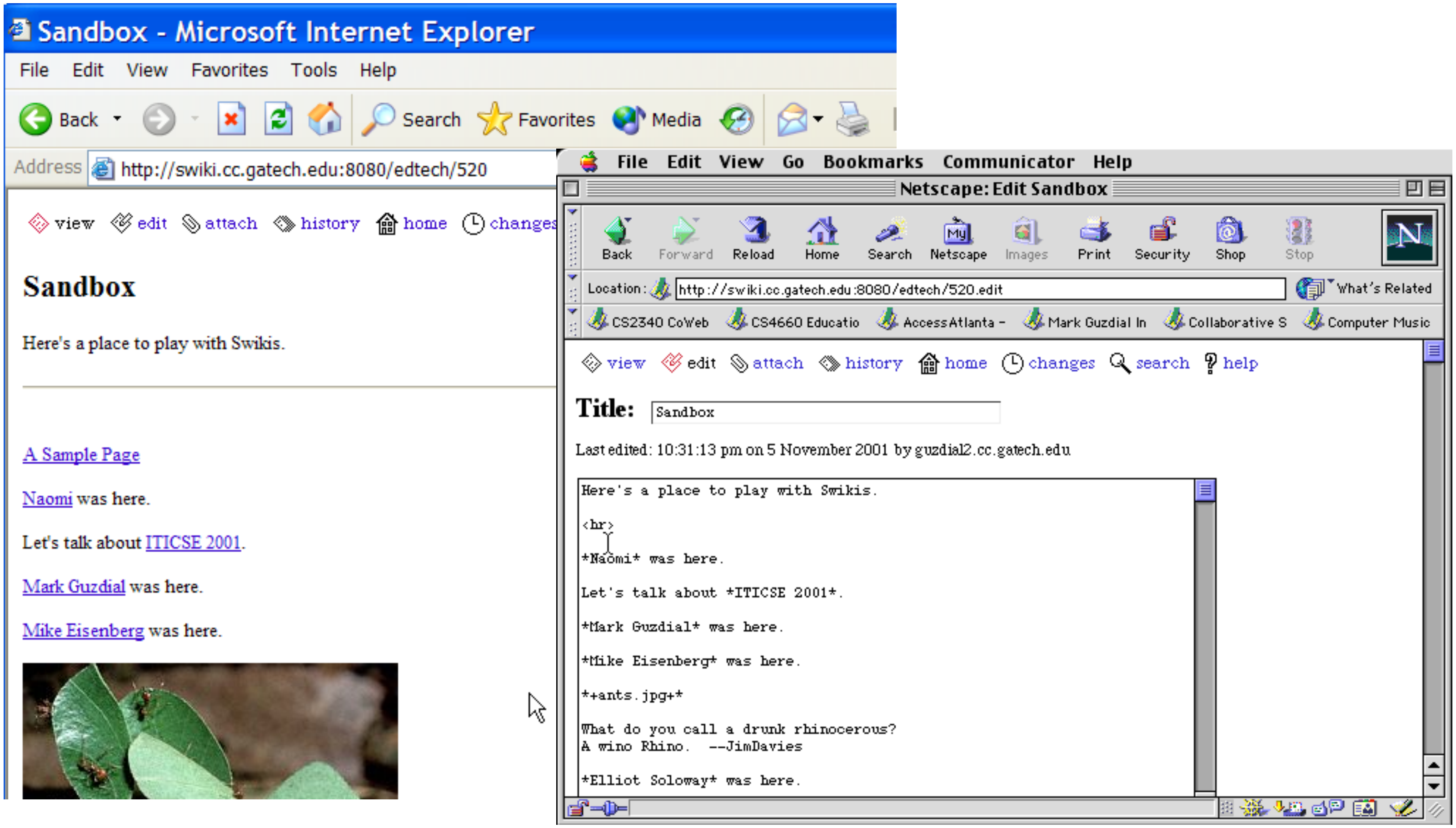
	ENROLLMENT	SUCCESS RATE
Gainesville's CSCI 1100		
Average 2000 – 2003	28	70.2%
Media Computation		
Summer 2003	9	77.8%
Fall 2003	39	84.6%
Spring 2004	22	77.3%
Summer 2004	11	90.9%



Our school's CS 0.5	Enrollment	Success Rate
Fall 2002	61	74.8%
Spring 2003	38	76.7%
Fall 2003	51	68.6%
Spring 2004	22	82.9 %
Fall 2004	15	93.3 %
Average "Old"	37	**75.9%**
New Spring 2005	18	94.4%
New Fall 2005	29	90.0%
New Fall 2006	42	76.2%
New Spring 2007	24	83.3%
Average "New"	28.3	**84.1%**

Table 1: Success rate with CS 0.5 before and with new approach. Averages weighted by enrollment.

Using the first Wiki for undergraduate learning


Sandbox - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
view edit attach history home changes
File Edit View Go Bookmarks Communicator Help
Netscape: Edit Sandbox
Back Forward Reload Home Search Netscape Images Print Security Shop Stop
Location:
What's Related
CS2340 CoWeb CS4660 Educatio AccessAtlanta - Mark Guzdial In Collaborative S Computer Music
view edit attach history home changes search help

http://swiki.cc.gatech.edu:8080/edtech/520

Sandbox

Here's a place to play with Swikis.

A Sample Page

Naomi was here.

Let's talk about ITICSE 2001.

Mark Guzdial was here.

Mike Eisenberg was here.

http://swiki.cc.gatech.edu:8080/edtech/520.edit

Title: Sandbox

Last edited: 10:31:13 pm on 5 November 2001 by guzdial2.cc.gatech.edu

```
Here's a place to play with Swikis.

<hr>

*Naomi* was here.

Let's talk about *ITICSE 2001*.

*Mark Guzdial* was here.

*Mike Eisenberg* was here.

*+ants.jpg+*

What do you call a drunk rhinocerous?
A wino Rhino.  --JimDavies

*Elliot Soloway* was here.
```

Role of the Homework "Galleries"

Q: What do you think about the homework galleries on the CoWeb?

Student 4: It's nice to see other people, like what they did with it… And there is no better feeling than getting something done and knowing that you've done it right.

Student 3 (Female, INTA): I don't ever look at it [the homework gallery] until after I'm done. I have a thing about not wanting to copy someone else's ideas. I just wish I had more time to play around with that and make neat effects. But JES [IDE created for this class] ***will be on my computer forever***, so… the nice thing about this class is that you could go as deep into the homework as you wanted. So, I'd turn it in, and then ***me and my roommate would do more after to see what we could do with it***.

Anecdotes from Engineering and Math

- On a mandatory assignment involving a math class studying results from Engineering students' simulations, 40% of math students accepted a *zero* rather than collaborate with engineers.
- We provided an Equation Editor in the CoWeb for an Engineering and a Math course to facilitate talking about equations. Not a single student even *tried* the Editor.
- Changed the focus of our research: Why not participate?

Competition

- Student quotes on “Why didn’t you participate in CoWeb?”

“1) didn't want to get railed 2) with the curve it is better when your peers do badly”

“since it is a curved class most people don’t want others to do well”

Note: Students claimed that the course grades were “curved” even when there was none.

Learned helplessness

- Student quotes:

  “I haven't posted about questions because I am confident that my answers are wrong.”

  “I thought I was the only one having problem understanding what was asked in the exam.”

  “Who am I to post answers?”

  “The overall environment for [this class] isn't a very help-oriented environment.”

Bottom line: Collaboration may not “just work” in Engineering/ Math/CS -- not without an explicit focus to make it work.

Story #2: Graphics Designers who program

- Brian Dorn studied graphics designers who program.
- Conducted a series of interviews and assessment activities.
- Found that these subjects want more computer science, but don't find courses (and most other resources) adequate (Dorn & Guzdial, ICER 2010)

- *P10: So, that was a really long way of saying* ***yes, I think that an academic study would make me a better programmer, but not by a whole lot.***

Who are graphics designers who program?

- Mostly arts/media trained.
- Don’t consider themselves programmers.
- But do some significant automation of their process.

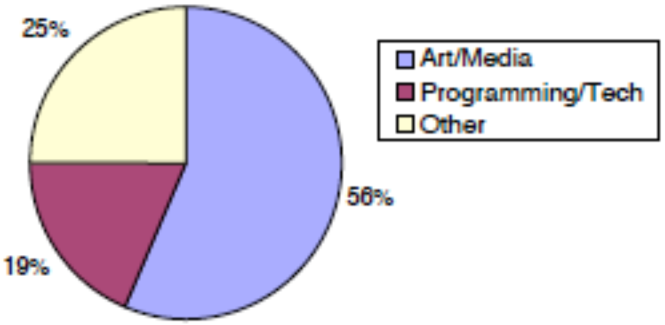
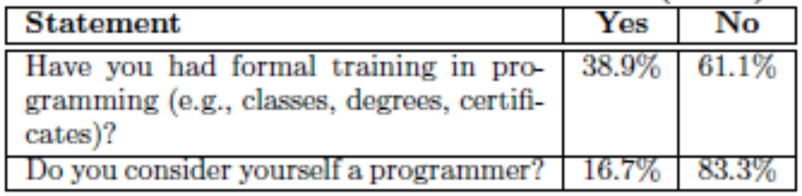


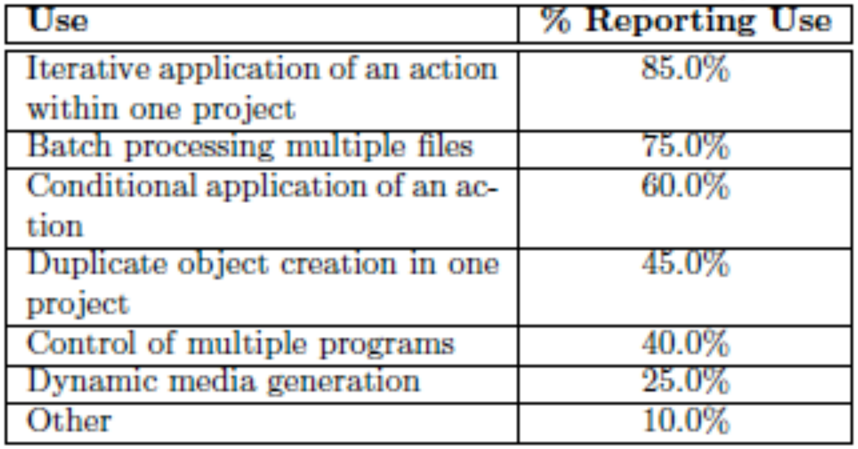
Figure 1: Area of Occupation (n=16)

Table 1: Coursework and Self-Affiliation (n=18)



Statement	Yes	No
Have you had formal training in programming (e.g., classes, degrees, certificates)?	38.9%	61.1%
Do you consider yourself a programmer?	16.7%	83.3%

Table 3: Uses for Automation (n=20)



Use	% Reporting Use
Iterative application of an action within one project	85.0%
Batch processing multiple files	75.0%
Conditional application of an action	60.0%
Duplicate object creation in one project	45.0%
Control of multiple programs	40.0%
Dynamic media generation	25.0%
Other	10.0%

Dorn & Guzdial, ICER 2006

Where are they getting their CS knowledge?

- Mostly on-line:
  - FAQs and other documentation
  - Books (when applicable)
  - Lots of examples and networking.
- Not so much classes

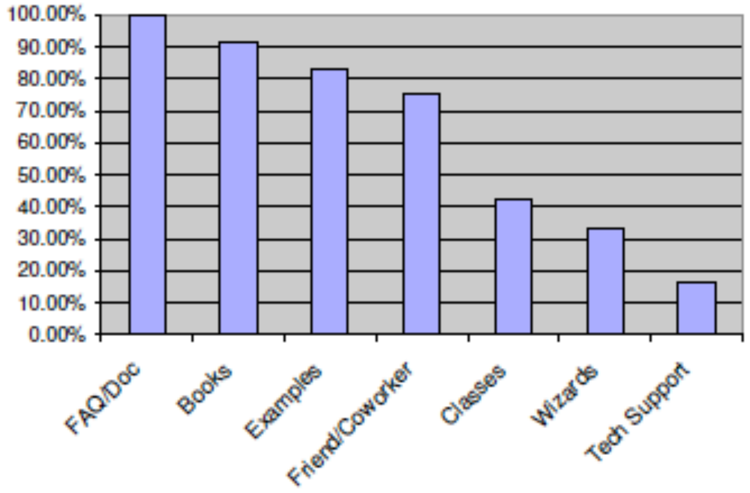
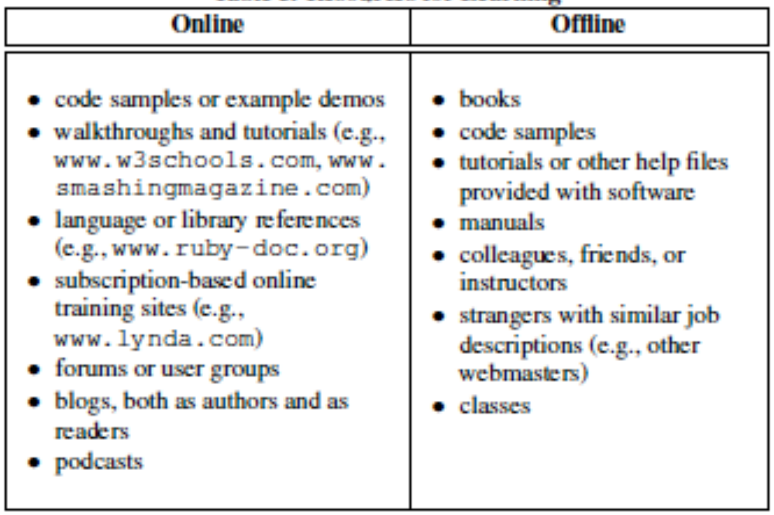


Figure 3. Percent of Participants Rating Resource as Likely or Very Likely to Use

Dorn & Guzdial, CHI 2010

Table 5. Resources for Learning



Online	Offline
• code samples or example demos • walkthroughs and tutorials (e.g., `www.w3schools.com`, `www.smashingmagazine.com`) • language or library references (e.g., `www.ruby-doc.org`) • subscription-based online training sites (e.g., `www.lynda.com`) • forums or user groups • blogs, both as authors and as readers • podcasts	• books • code samples • tutorials or other help files provided with software • manuals • colleagues, friends, or instructors • strangers with similar job descriptions (e.g., other webmasters) • classes

What do software engineers do?
Answer: The Boring Stuff.

- P2: I was able to take different samples from different places and instead of just being let's say an MIS major, or computer science major, you know it's—**you're not going to be front-end anything with computer science. You're going to be back-end everything.**

- P4: I think as **a front-end developer**, you focus more on the design and the usability, and you're focusing more on the audience. And then on the **back-end** I think you're focused on more, these are like the software developers. **And they're programming something, and they don't really see what it's gonna look like; they're just making it work.**

Who is in CS?

- *Like Yardi and Bruckman (ICER 2007), participants held negative stereotypes of those in CS:*

- P2: I went to a meeting for some kind of programmers, something or other. **And they were OLD, and they were nerdy, and they were boring!** And I'm like, this is not my personality. Like I can't work with people like that. And they worked at like IBM, or places like that. They've been doing, they were working with Pascal. And I didn't…**I couldn't see myself in that lifestyle for that long.**

- P5: I don't know a whole ton of programmers, but the ones I know, **they enjoy seeing them type up all these numbers and stuff and what it makes things do.** Um, whereas I just do it, to get it done and to get paid. To be honest. The design aspect is what really interests me a lot more.

Georgia Tech College of Computing

Why don’t they take CS classes?

- P7**: I started out in computer science, but didn't like it at all.** The fact that I wasn't learning anything new. I took an intro to programming course, and then I talked to some other people in the program and it was all repetition and I guess there wasn't any really new**. So you weren't really learning any concepts. You were learning the languages, and I didn't like that at all. So that's why I left…**
- *Do* we just teach *languages*?
  Why don’t they *see* the *concepts*?

They are not afraid of coding

- “What interests you about web design?”
- P12: The coding! I don't like to code. But the things that the code can do is amazing, like you can come up with this and voila, you know, it's there. Javascript for one. The plugins and stuff. I think that's very interesting, intriguing and stuff. Because I mean like the code is just, there's so much you can do with code and stuff. It's just like wow.

They want to know more

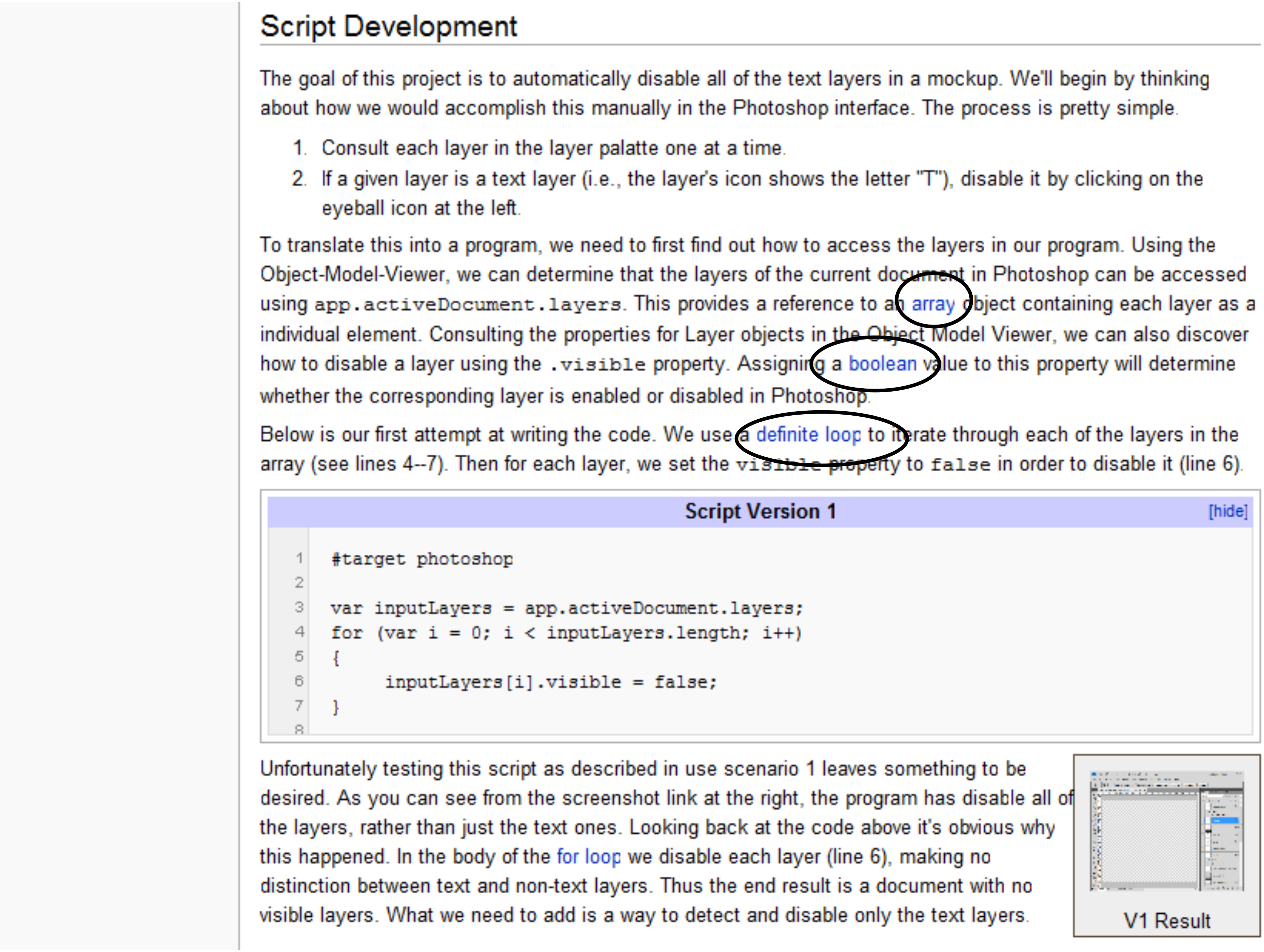
- P1: So I mean technology changes. So what I am ideally looking to focus on are like the foundation. The things that change less, you know what I'm saying? **Like computer science um, theory**, you know I'm saying I mean? That kind of like, it's applicable to what I do, and it's not so constantly shifting.

- P10: I was the kind of programmer that could make stuff work. But I didn't really have solid understandings. At one point I picked up **a book on design patterns** and I looked at it, and I was like that's really, that's really interesting…So I was like well **I wanna keep doing that because it made me a better programmer. And it was more fun to program, and it was more thought provoking.**

They’re Lost without Initial Knowledge

- Learning less than they might because of a lack of deep knowledge.
  - For example: Exploring code by searching Google for function and variable names.
  - Learning about Java when programming in JavaScript

- **Brian’s experiment:** Given a case library **with** conceptual information vs. a code repository **alone,** what gets learned, used, and liked? (ICER 2011)



Script Development

The goal of this project is to automatically disable all of the text layers in a mockup. We'll begin by thinking about how we would accomplish this manually in the Photoshop interface. The process is pretty simple.

1. Consult each layer in the layer palatte one at a time.
2. If a given layer is a text layer (i.e., the layer's icon shows the letter "T"), disable it by clicking on the eyeball icon at the left.

To translate this into a program, we need to first find out how to access the layers in our program. Using the Object-Model-Viewer, we can determine that the layers of the current document in Photoshop can be accessed using `app.activeDocument.layers`. This provides a reference to an array object containing each layer as a individual element. Consulting the properties for Layer objects in the Object Model Viewer, we can also discover how to disable a layer using the `.visible` property. Assigning a boolean value to this property will determine whether the corresponding layer is enabled or disabled in Photoshop.

Below is our first attempt at writing the code. We use a definite loop to iterate through each of the layers in the array (see lines 4--7). Then for each layer, we set the `visible` property to `false` in order to disable it (line 6).

Script Version 1 [hide]

```
1  #target photoshop
2
3  var inputLayers = app.activeDocument.layers;
4  for (var i = 0; i < inputLayers.length; i++)
5  {
6       inputLayers[i].visible = false;
7  }
8
```

Unfortunately testing this script as described in use scenario 1 leaves something to be desired. As you can see from the screenshot link at the right, the program has disable all of the layers, rather than just the text ones. Looking back at the code above it's obvious why this happened. In the body of the for loop we disable each layer (line 6), making no distinction between text and non-text layers. Thus the end result is a document with no visible layers. What we need to add is a way to detect and disable only the text layers.

V1 Result

Bottomline: Cases work

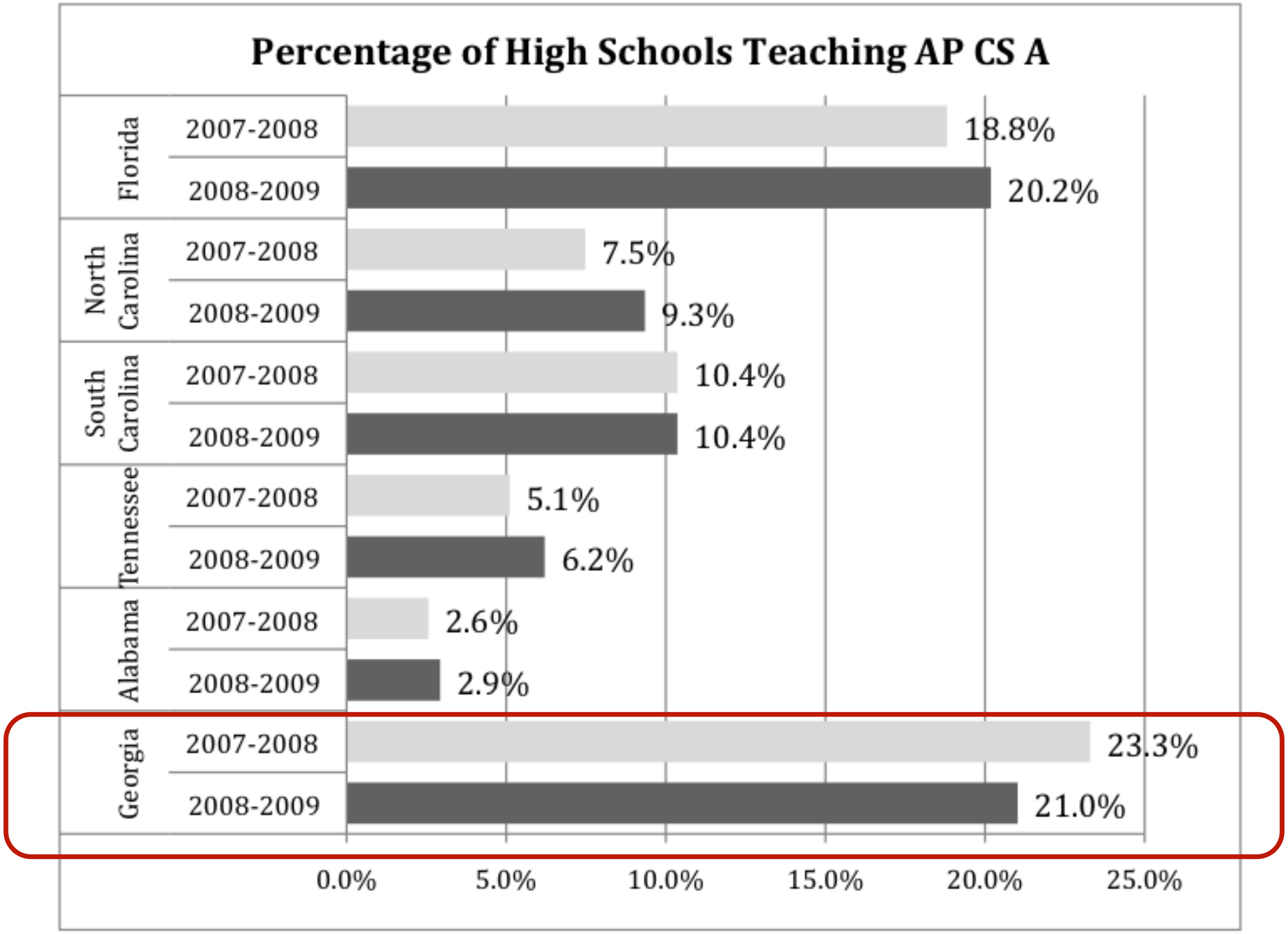
- They *like* the cases.
- They *code* the same.
- Case-users *learn* the concepts, while repository users do not.

- Suggests how we might help non-CS professionals who discover computing late.

Story #3: High School teachers need a new pedagogy

- For the most part, we teach CS as apprenticeship.
  - We model (lecture), students practice (code), we try to coach.
- Practice is *always* important.
  Do we rely on it too much?
  - Are we more like STEM or Architecture?
  - Claim: No other science and engineering field teaches so much through practice.
- Result: CS pedagogy is time-consuming.

Problem:
Too little secondary school CS



What makes a "Computer Science" high school teacher?

- High school teachers who self-identify as a *kind* of teacher are more likely to be retained, seek professional development, and join a professional community.
- But most teacher identity is tied to certification.
  - CS teacher certification does not exist in most states.
- How do teachers of CS identify themselves? How do they develop an identity as a CS teacher? (Lijun Ni, 2011 thesis)

Teachers feel a need for more training

- *[Becky]: "I struggle with giving everyone the material and being able to explain it to everyone… I struggle with how to be creative with the programming. I have a problem with trying to make the programs have meaning to them… It is hard to teach. It's hard knowing how to teach it, how to give it to them… It's hard to explain. When I look at kid's codes, they think I should know it… They think that I should know it as soon as I look at it. For the longest time I thought I should, but I don't have to. I have to study it just like they do. So, I would like some training."*

Easier with Support

- Confidence: More confident in teaching Math

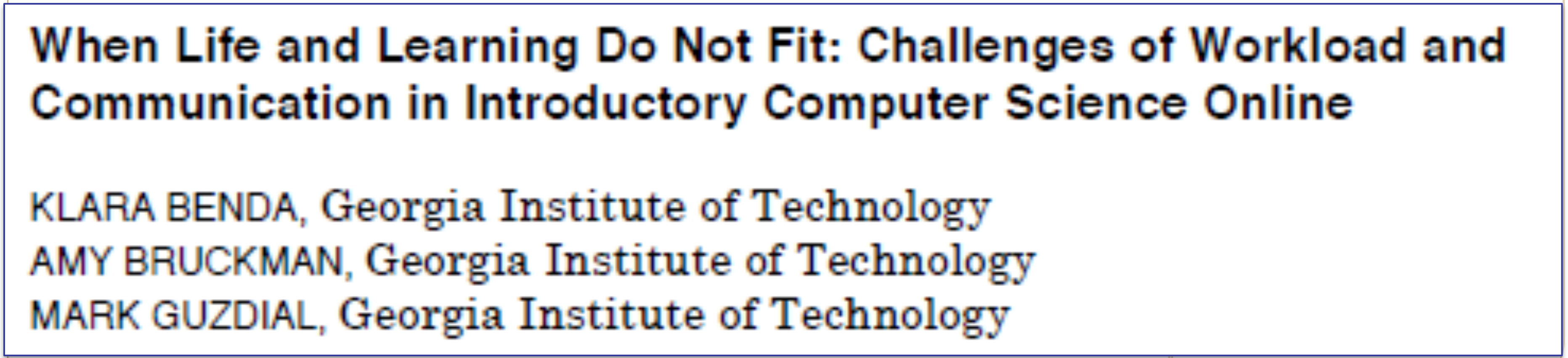
  [John]: "I still think I'm a better Math teacher, just because I've had so much support. Whenever I have problems, I can talk with the people that I work with, most of who have taught for many years in Math… With Computer Science, I've got nobody to talk to."

They may find CS inaccessible

- *[May]: "I think,* computer science is more for really, really smart people. *I'm not saying I'm smart, but I'm thinking that if I have to go take this Computer Science degree, that it's going to be really hard, because it's going to ask a lot of programming questions, syntax questions. I think computer science is a much higher level…* When I say computing, I think of computing as being able to operate the computer, … *I believe that most students can successfully take and complete Computing in the Modern World, but* it takes a little higher level of intelligence to complete the Introduction to Programming

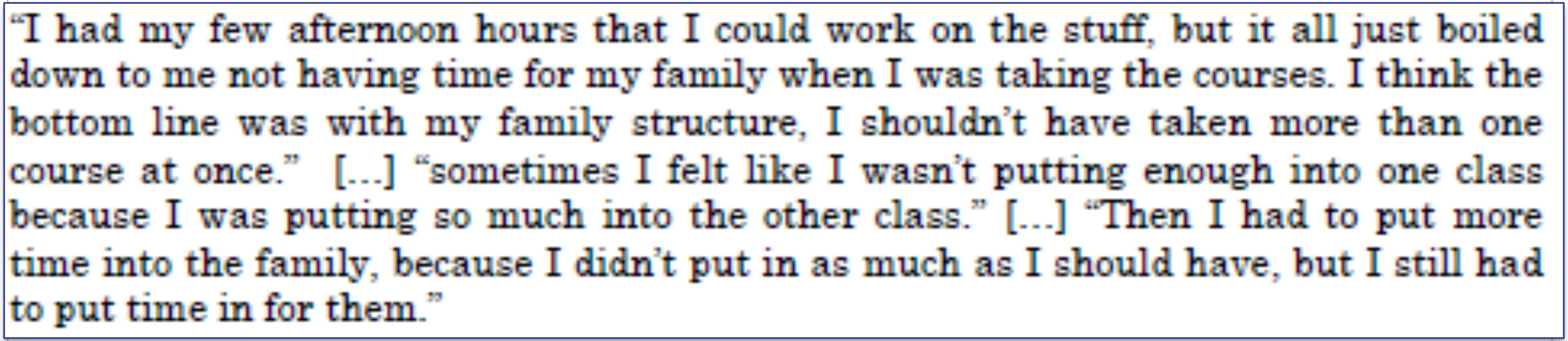
How do we teach working high school teachers?

- Study of adult/professional students in CS classes.
  - They don’t have the time to spend hours in front of the IDE.
  - Lacking background, e.g., in mathematics.
  - They get stymied by small errors.

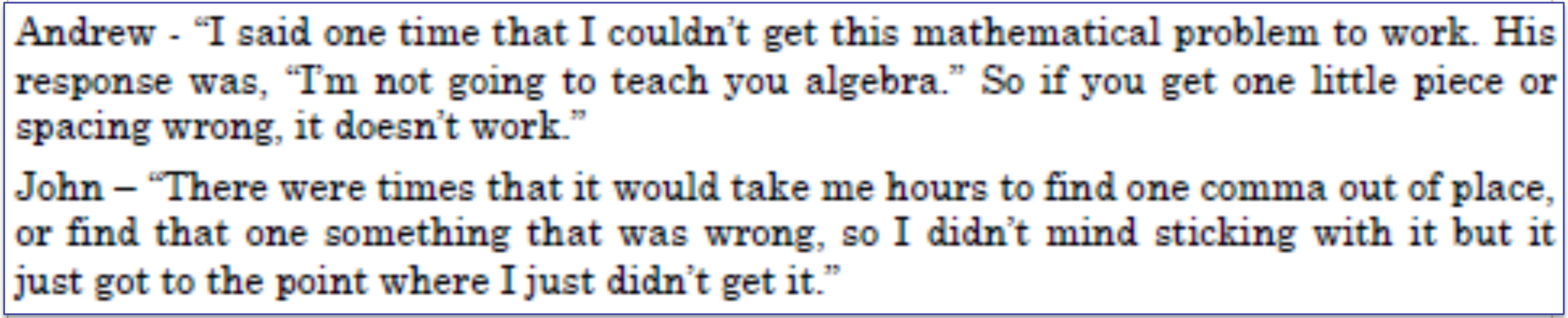


When Life and Learning Do Not Fit: Challenges of Workload and Communication in Introductory Computer Science Online

KLARA BENDA, Georgia Institute of Technology
AMY BRUCKMAN, Georgia Institute of Technology
MARK GUZDIAL, Georgia Institute of Technology



"I had my few afternoon hours that I could work on the stuff, but it all just boiled down to me not having time for my family when I was taking the courses. I think the bottom line was with my family structure, I shouldn't have taken more than one course at once." [...] "sometimes I felt like I wasn't putting enough into one class because I was putting so much into the other class." [...] "Then I had to put more time into the family, because I didn't put in as much as I should have, but I still had to put time in for them."

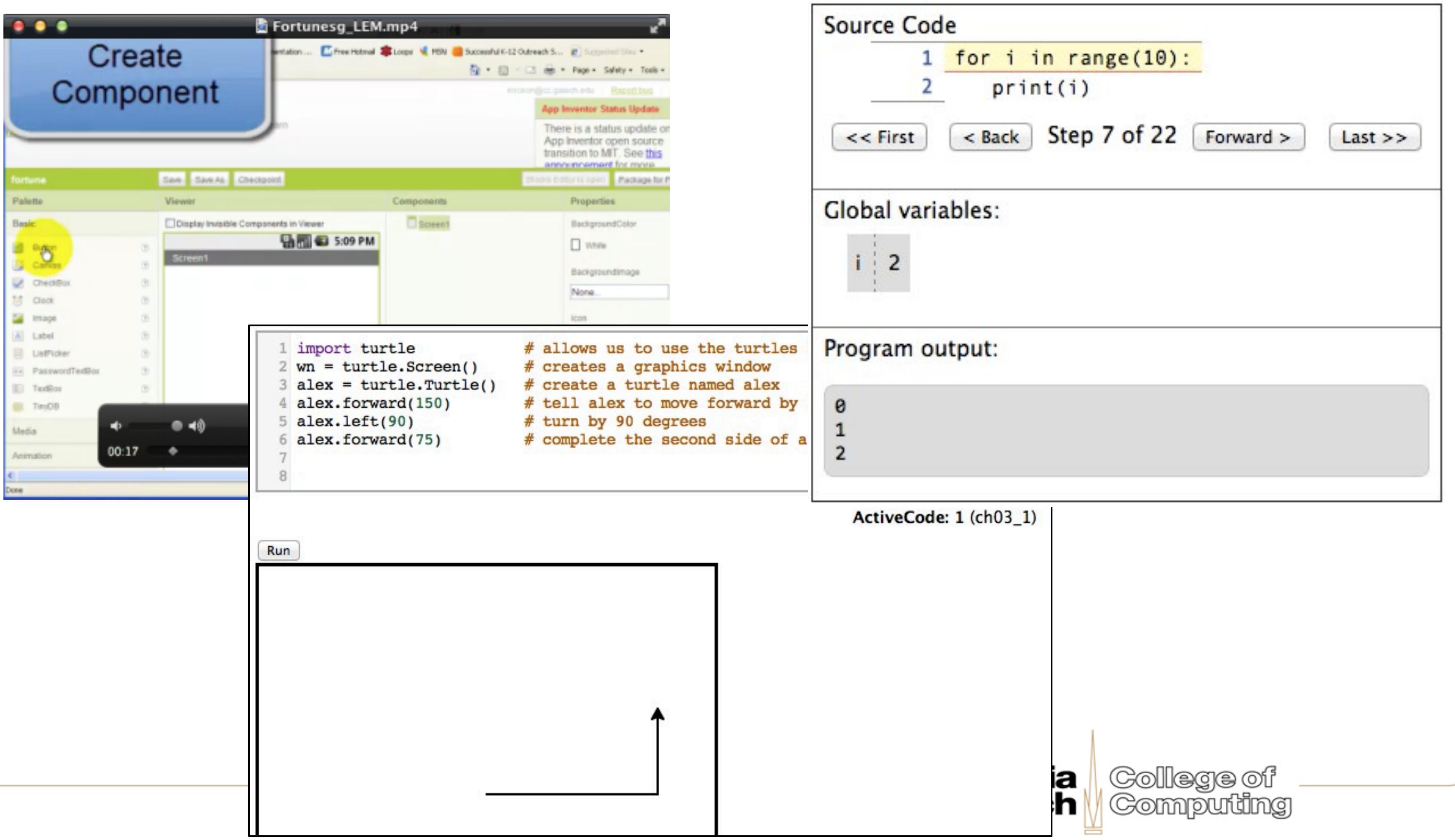


Andrew - "I said one time that I couldn't get this mathematical problem to work. His response was, "I'm not going to teach you algebra." So if you get one little piece or spacing wrong, it doesn't work."

John – "There were times that it would take me hours to find one comma out of place, or find that one something that was wrong, so I didn't mind sticking with it but it just got to the point where I just didn't get it."

Trying to teach CS more Efficiently

- Using a model of a book, but with live code and Ed Psych principles:



```
import turtle              # allows us to use the turtles
wn = turtle.Screen()       # creates a graphics window
alex = turtle.Turtle()     # create a turtle named alex
alex.forward(150)          # tell alex to move forward by
alex.left(90)              # turn by 90 degrees
alex.forward(75)           # complete the second side of a
```

ActiveCode: 1 (ch03_1)

Source Code

```
for i in range(10):
    print(i)
```

Global variables:

i	2

Program output:

```
0
1
2
```

Conclusions

- Computing is important to let everyone create.
  It's the job of computer scientists to give it to everyone

- Story #1: Non-CS Major students succeed with contexts.

- Story #2: End-user programmers want what CS has to offer, and there are more of them than there are professional software developers. But they don't know enough CS to teach themselves effectively.

- Story #3: Teachers don't know enough computer science, want more training, but existing methods don't work well for them.

With thanks to our supporters

- US National Science Foundation
  - Statewide BPC Alliance: Project “Georgia Computes!” http://www.gacomputes.org
  - CCLI and CPATH Grants, and now CE21 to produce new media
- Microsoft Research
- Georgia Tech's College of Computing
- Georgia’s Department of Education
- GVU Center,
- GT President's Undergraduate Research Award,
- Toyota Foundation

Thank you!

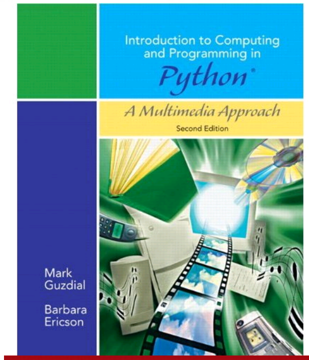
- http://www.cc.gatech.edu/~mark.guzdial

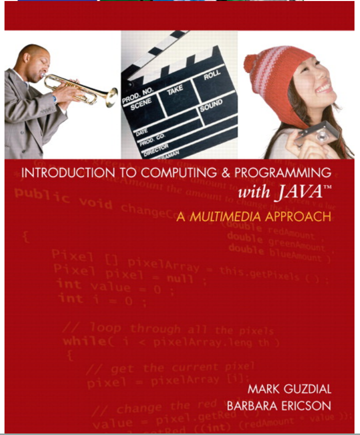
  http://home.cc.gatech.edu/csl

  http://www.georgiacomputes.org

For more on CSLearning4U:

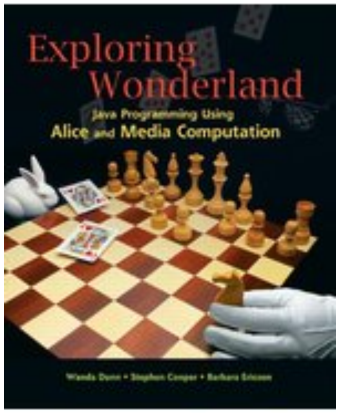
- http://home.cc.gatech.edu/csl/CSLearning4U

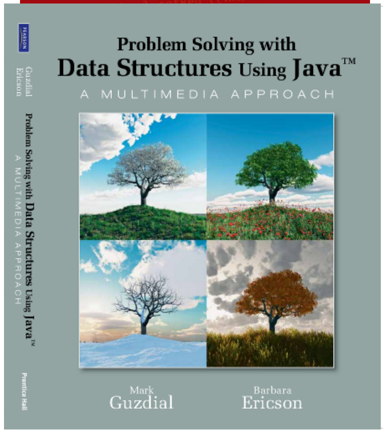




For more on MediaComp approach:

- http://www.mediacomputation.org


Exploring Wonderland
Java Programming Using
Alice and Media Computation
Wanda Dann • Stephen Cooper • Barbara Ericson



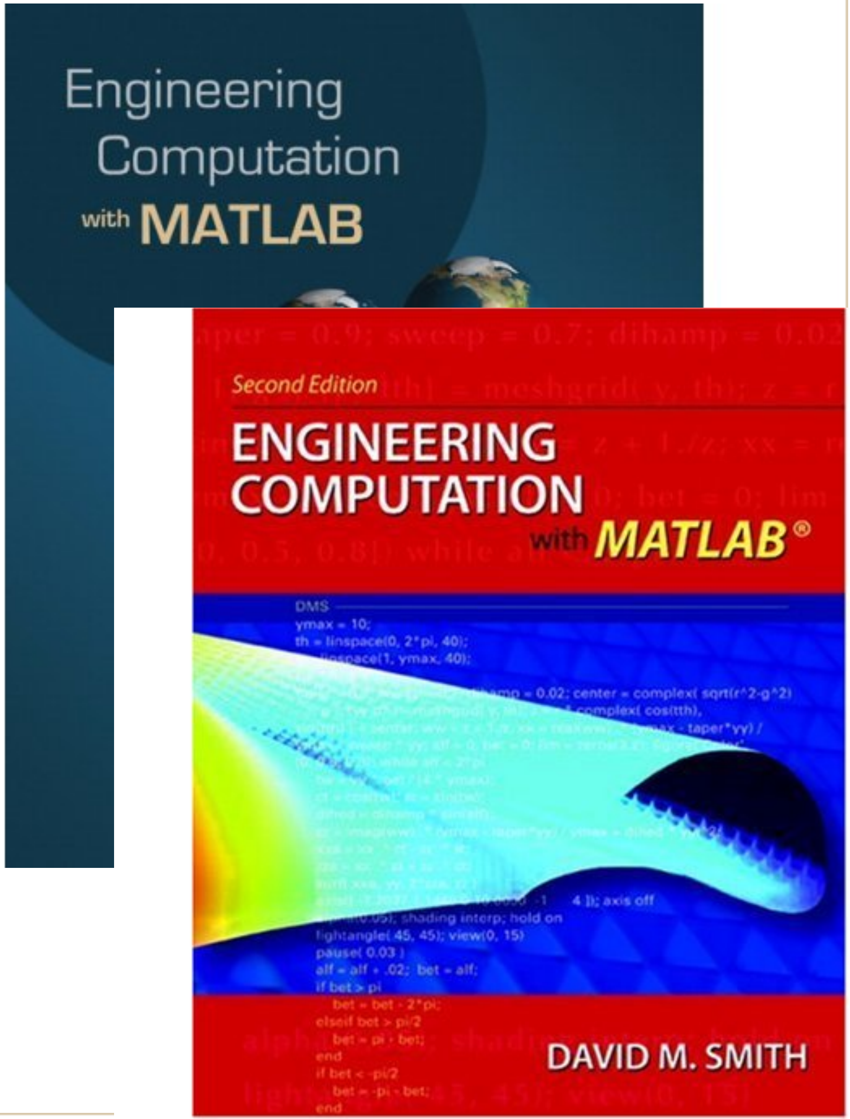
Spare Slides

Can we teach more of CS *more efficiently*?

- Matt Jadud [2006] (and others) has shown us how small Java errors can lead to an enormous waste of time.
  - Can we teach about variables, behavior, how loops work, how conditionals behave – without a half hour of “where goes the semi-colon”?
- Can we reduce the wasted time?
  - Analogy: Does coursework in a foreign language make it easier to be immersed in the new language, or is immersion the only way to learn?
- New NSF CE21 Project: CSLearning4U http://home.cc.gatech.edu/csl/CSLearning4u

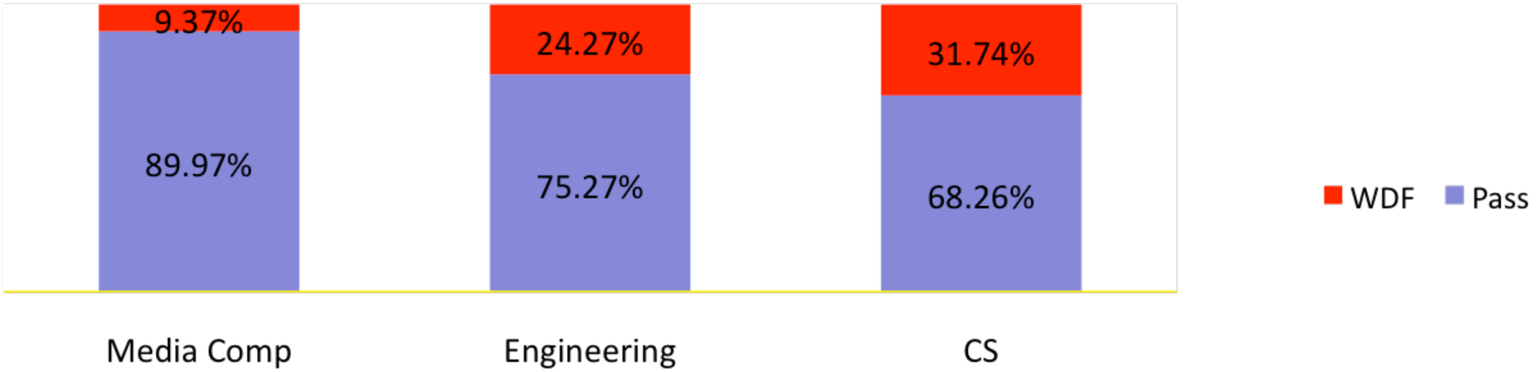
Introducing Computing in an Engineering Context

- Developed in collaboration with Civil, Mechanical, and Aerospace Engineering.
- Uses Engineering problems and MATLAB
- Covers traditional CS1 topics
- Among our 3 CS1’s, these are the first students to program outside of class.
- The success rate in this class also rose compared to all-in-one.


Engineering Computation with MATLAB
Second Edition
ENGINEERING COMPUTATION with MATLAB®
DAVID M. SMITH

Comparing Spring 2004

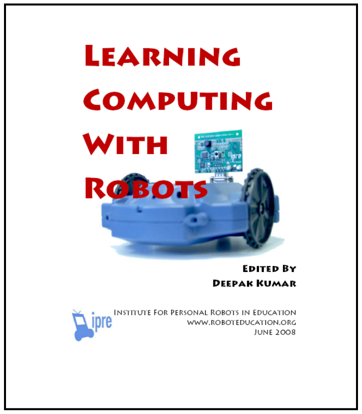
CS for Engineers: ~1200 students/semester
Media Comp: ~300 students/semester
CS for CS majors: ~150 students/semester



A Context for CS1 for CS majors: Robotics

- Microsoft Research has funded the Institute for Personal Robotics in Education
  - Leads: Tucker Balch, Deepak Kumar, Doug Blank
  - Joint between Bryn Mawr College and Georgia Tech
  - http://www.roboteducation.org

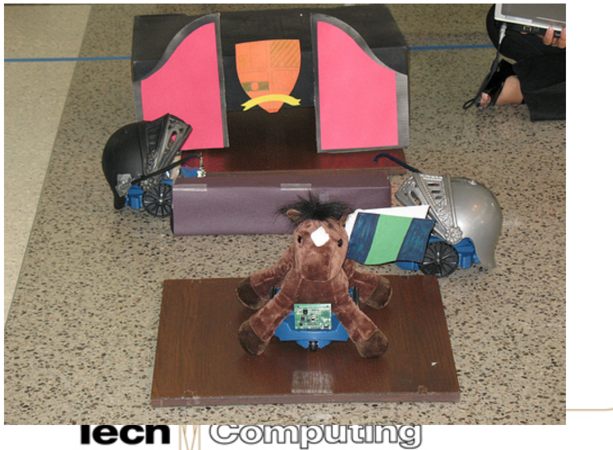
- Developing a CS1 with robotics as the context.



LEARNING COMPUTING WITH ROBOTS

EDITED BY
DEEPAK KUMAR

INSTITUTE FOR PERSONAL ROBOTS IN EDUCATION
WWW.ROBOTEDUCATION.ORG
JUNE 2008



Results at University of California, San Diego

Simon, Kinnunen, Porter, Zaskis, ACM ITICSE 2010

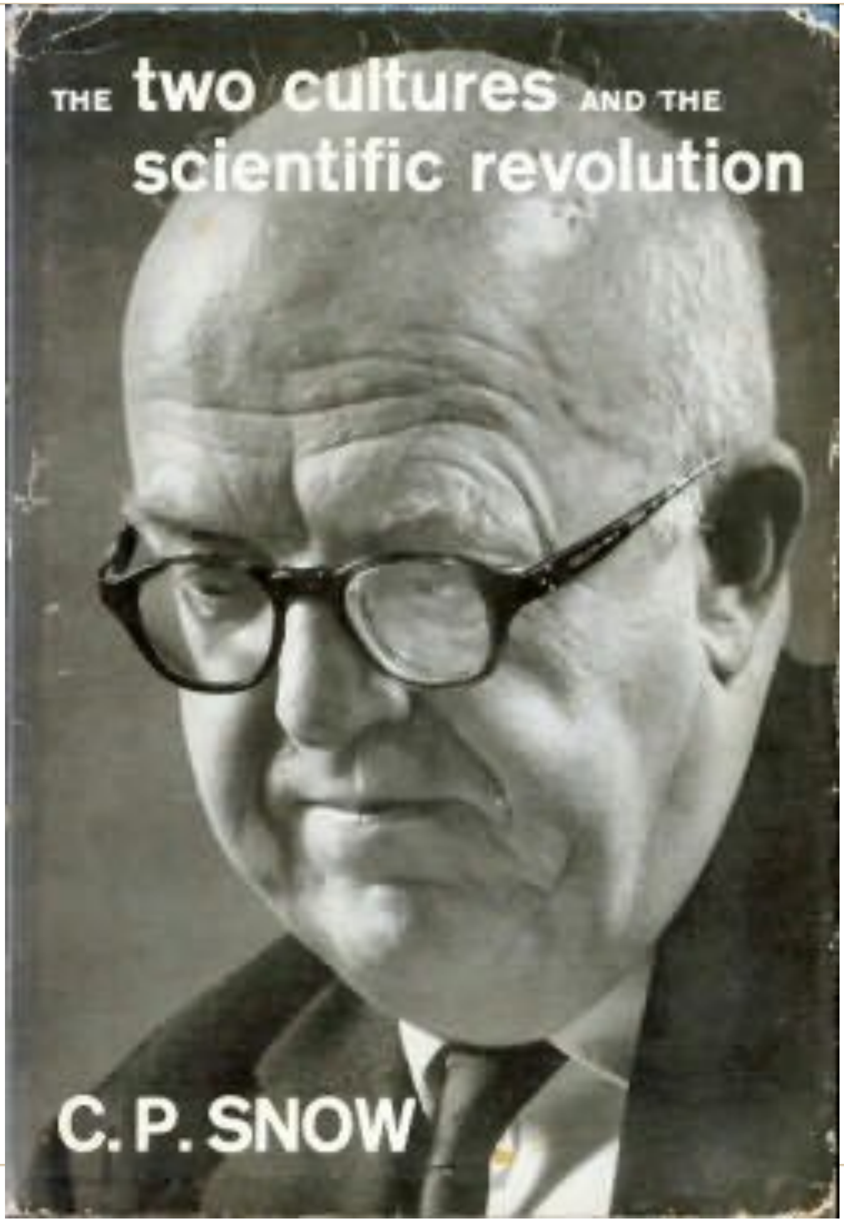
- Using Java Media Computation as normal CS1 for CS majors at a research university.
- Did extensive data collection last semester before switching to Media Computation.
- Been following two cohorts of CS1 students for comparison.

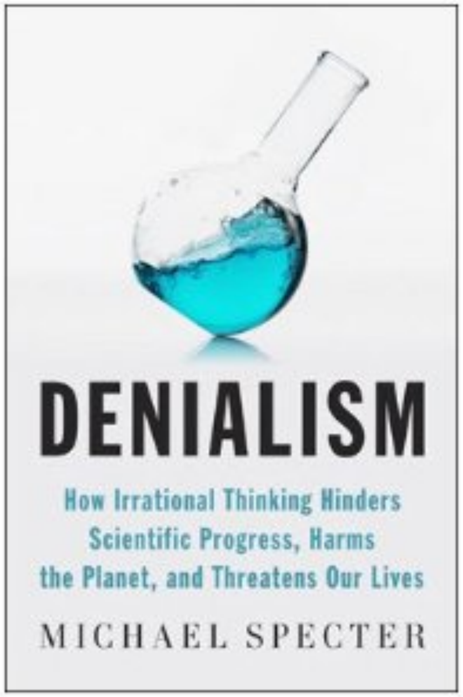
Findings:

- MediaComp has more focus on problem-solving, less on language.

- MediaComp students have higher pass rates and retention rates one year later

The Two Cultures




DENIALISM
How Irrational Thinking Hinders
Scientific Progress, Harms
the Planet, and Threatens Our Lives
MICHAEL SPECTER

Does it have to be *programming?*

- **Elias**: If the computers, together with sufficiently ingenious languages and programming systems, are capable of doing everything that Professor Perlis describes—and I believe they are (and more)—then ***they should be ingenious enough to do it without the human symbiote being obliged to perform the mechanical chores*** which are a huge part of current programming effort, and which are a large part of what must now be taught in the introductory course that he proposes.

Why *programming*

- ***Licklider***: Peter, I think the first apes who tried to talk with one another decided that learning language was a dreadful bore…But some people write poetry in the language we speak.

- ***Perlis***: The purpose of a course in programming is to teach people how to construct and analyze processes…A course in programming is concerned with *abstraction*: the abstraction of constructing, analyzing, and describing processes…***The point is to make the students construct complex processes out of simpler ones….A properly designed programming course will develop these abilities better than any other course***.